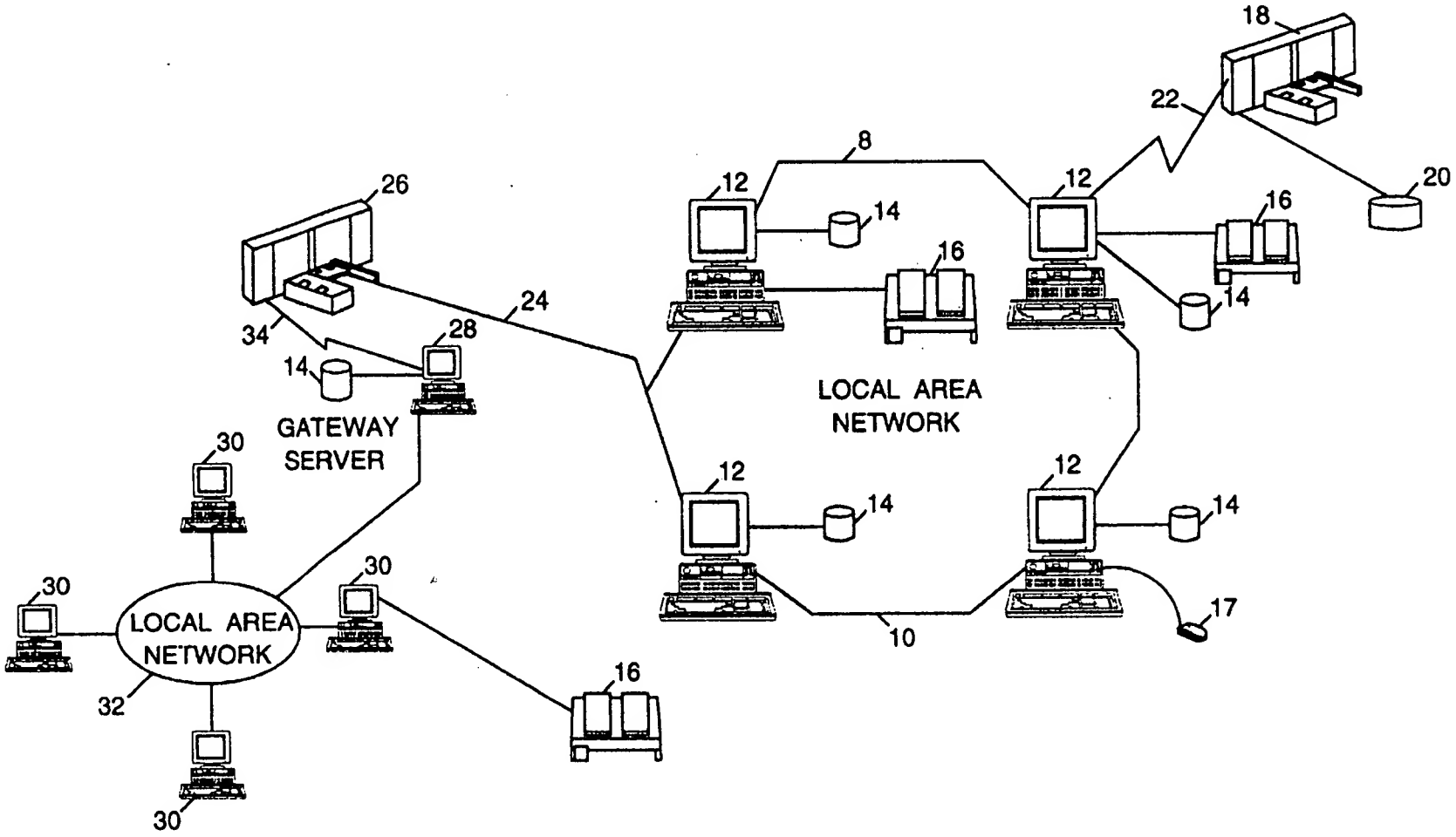
18
22
20
8
12
12
16
14
16
14
26
24
34
28
14
GATEWAY
SERVER
LOCAL AREA
NETWORK
30
12
12
14
14
30
30
LOCAL AREA
NETWORK
17
10
16
32
30

FIG. 1

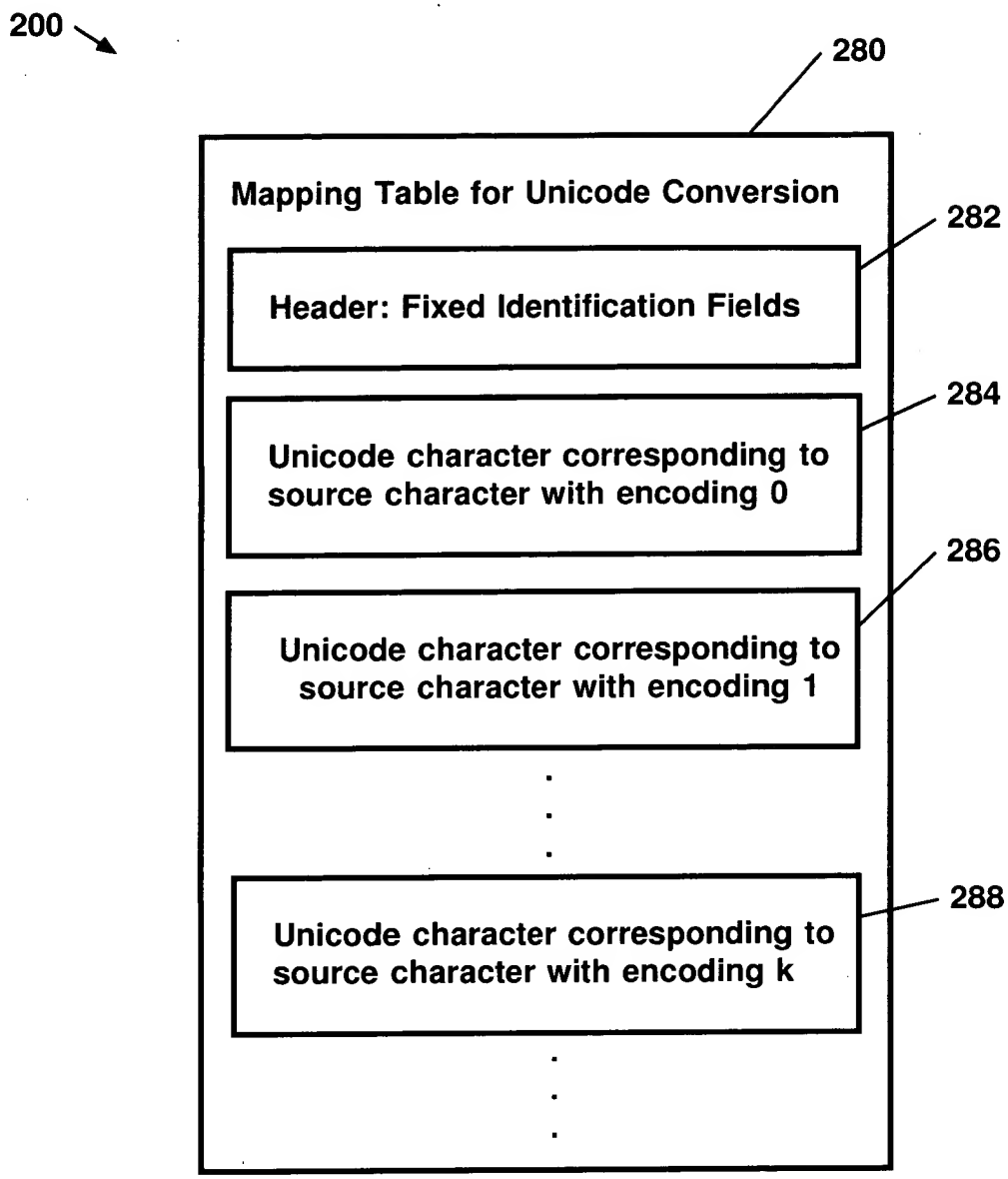
200
280
Mapping Table for Unicode Conversion
282
Header: Fixed Identification Fields
284
Unicode character corresponding to source character with encoding 0
286
Unicode character corresponding to source character with encoding 1
288
Unicode character corresponding to source character with encoding k

Fig. 2

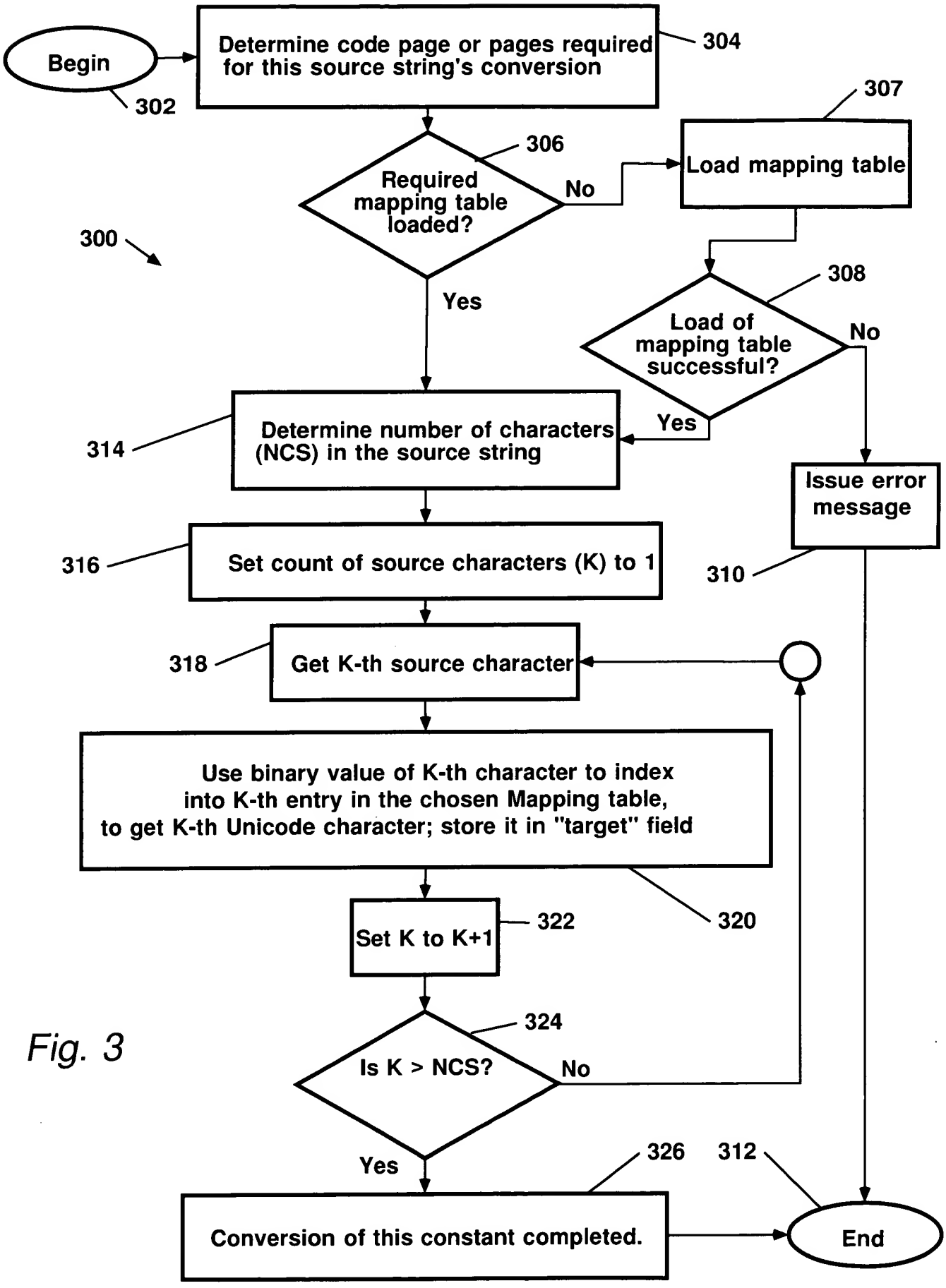
Begin
302
Determine code page or pages required for this source string's conversion
304
Required mapping table loaded?
306
No
Load mapping table
307
Load of mapping table successful?
308
No
Yes
Yes
300
Determine number of characters (NCS) in the source string
314
Issue error message
310
Set count of source characters (K) to 1
316
Get K-th source character
318
Use binary value of K-th character to index into K-th entry in the chosen Mapping table, to get K-th Unicode character; store it in "target" field
320
Set K to K+1
322
Is K > NCS?
324
No
Yes
Conversion of this constant completed.
326
End
312

Fig. 3

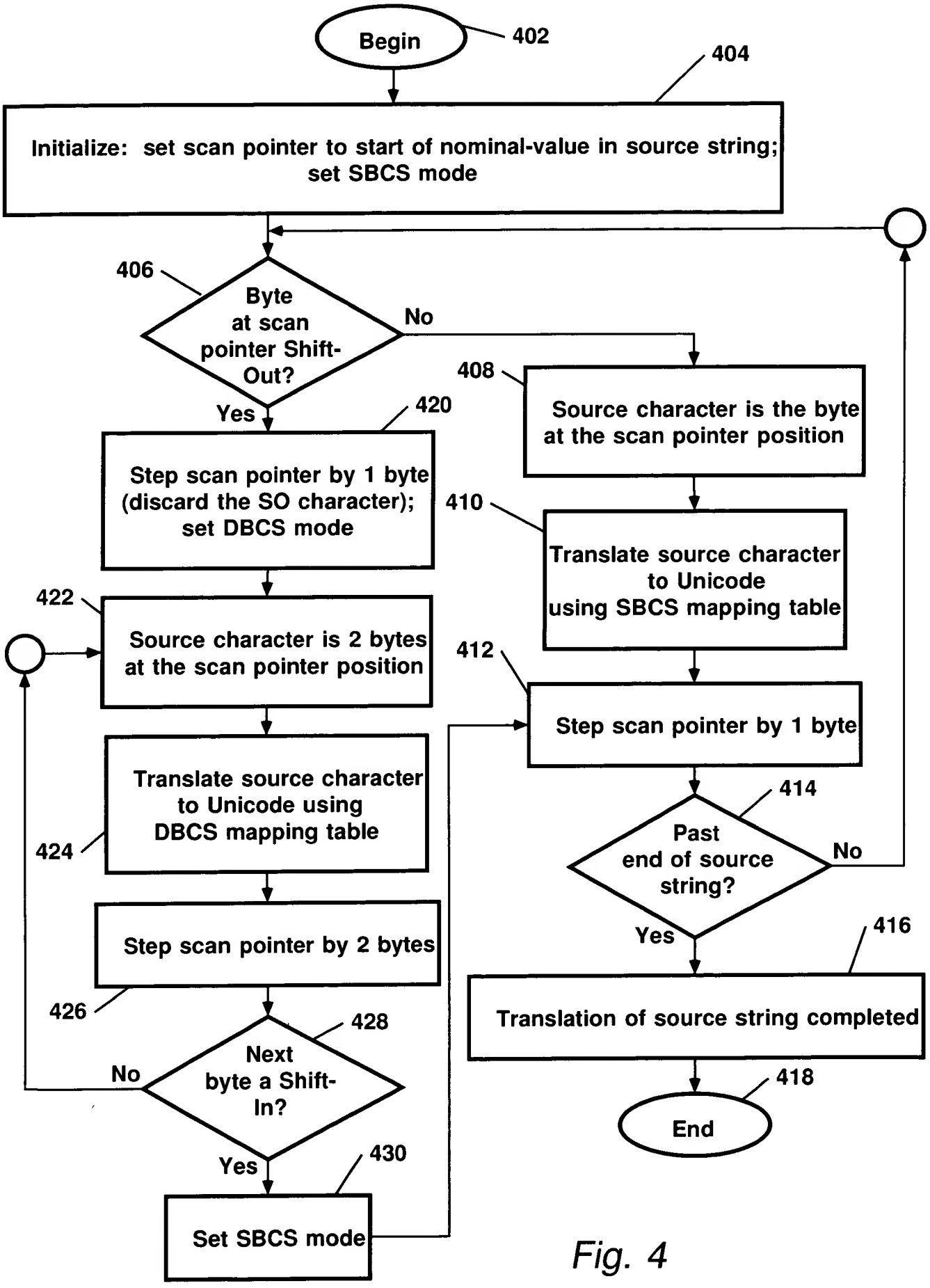
Begin
402
404
Initialize: set scan pointer to start of nominal-value in source string; set SBCS mode
406
Byte at scan pointer Shift-Out?
No
Yes
408
Source character is the byte at the scan pointer position
420
Step scan pointer by 1 byte (discard the SO character); set DBCS mode
410
Translate source character to Unicode using SBCS mapping table
422
Source character is 2 bytes at the scan pointer position
412
Step scan pointer by 1 byte
Translate source character to Unicode using DBCS mapping table
424
414
Past end of source string?
No
Yes
Step scan pointer by 2 bytes
426
416
Translation of source string completed
428
Next byte a Shift-In?
No
418
End
Yes
430
Set SBCS mode

Fig. 4

500

```
         Title 'DCU — a macro to generate Unicode constants'
         Macro
&L       DCU   &A,&Pair=Yes,&CodePage=500
.*
.*       Expected argument: an apostrophe-delimited string of one or
.*       more EBCDIC characters, with paired internal apostrophes and
.*       ampersands. The pairing is preserved in the output string if
.*       &Pair=Yes, and is not if &Pair=No.
.*       Initial limitation: max of 63 characters in quoted argument,
.*       except for paired characters if &Pair=No.
.*-------------------------------------------------------------------*
.*       Declare variables used internally                           *
.*-------------------------------------------------------------------*
         LclC  &M(256)          Mapping and validation table
         LclC  &V               Valid EBCDIC characters
         LclC  &R               Result Unicode string
         LclB  &P               True if '& pairs retained in output
         LclA  &J               Counter
         LclA  &N               Temp
.*-------------------------------------------------------------------*
.*       Validate macro arguments                                    *
.*-------------------------------------------------------------------*
         AIf   (N'&SysList gt 0).V1 Check for argument
         MNote 8,'DCU — No argument.'
         MExit
.*
.V1      AIf   (N'&SysList lt 2).V2 Check single argument
         MNote 8,'DCU — More than one argument.'
         MExit
.*
.V2      AIf   (K'&A ge 3).V3
         MNote 8,'DCU — argument too short, or badly formed.'
         MExit
.*
.V3      AIf   ('&A'(1,1) eq "" and '&A'(K'&A,1) eq "").V4
         MNote 8,'DCU — argument not properly quoted.'
         MExit
.*
.V4      AIf   ('YES' eq (Upper '&Pair')).V5  Check if pairing wanted
         AIf   ('NO' eq (Upper '&Pair')).V6   Check if no pairing
         MNote 8,'DCU — invalid value of &&Pair.'
         MExit
.*
.V5      ANop  ,
&P       SetB  1                Indicate no pairing of '& in output
.*
```

Fig. 5

600

```
.*
.V6     ANop  ,
        AIf   ('&CodePage' eq '500').V7   Check code page
        MNot  8,'DCU — Code Page &Cod  Page not supported yet.'
        MExit
.*
.V7     ANop  ,
.*-------------------------------------------------------------*
.*      Arguments validated. Set SBCS and Unicode character sets *
.*-------------------------------------------------------------*
.VX     ANop  ,                 Set up mapping table
&J      SetA  &J+1
&M(&J)  SetC  ''                Initialize to null
        AIf   (&J lt 256).VX    Loop for all 256 code points
.*
&V      SetC  '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrs*
              tuvwxyz_@#$%&&*()-+=,./:;"<>"? '
.*
.*      The following is the conversion table from CCSID 500 to Unicode
.*
&U      SetC  '3031323334353637383941424344454647484 94A4B4C4D4E4F50515*
              2535455565758595A6162636465666768696A6B6C6D6E6F707172737*
              475767778797A5F4023242526262A28292D2B3D2C2E2F3A3B273C3E2*
              23F20'
.*
.*      Note: Conditional-assembly string constants require paired
.*      apostrophes and ampersands; ampersands are not reduced to a
.*      single character internally. Thus, the encoding for & appears
.*      twice in the &U encoding string above.
.*
&J      SetA  1
.*-------------------------------------------------------------*
.*      Build the EBCDIC-to-Unicode mapping table               *
.*-------------------------------------------------------------*
.VY     ANop
&C      SetC  (Double '&V'(&J,1)) Pick character from valid string
&C      SetC  'C"&C"'           Character in self-defining term
&N      SetA  &C+1               Convert to numeric
&M(&N)  SetC  '&U'(2*&J-1,2)     Put Unicode digits in mapping table
&J      SetA  &J+1               Increment &J
        AIf   (&J le K'&V).VY    Set up all valid encodings
```

Fig. 6

```
.*-------------------------------------------------------------*
.*    Convert each SBCS argument character to Unicod  equivalent  *
.*-------------------------------------------------------------*
&J     SetA  2              Start after initial apostrophe
.*
.Z     ANop  ,              Head of translation loop
&C     SetC  '&A'(&J,1)       &J-th character from argument
700 ↘  AIf   ('&C' ne '''' and '&C' ne '&&'(1,1)).Z1 Is it '& ?
       AIf   (&P).Z1        Have '&, is pairing wanted?
&J     SetA  &J+1             No pairing, step input by one
.Z1    ANop  ,
&C     SetC  (Double '&C')     Pair '& for self-defining term
&C     SetC  'C''&C'''         Change to arithmetic value
&N     SetA  &C+1              Convert to numeric
&C     SetC  '&M(&N)'          Get Unicode mapping
       AIf   ('&C' ne '').Z2   Validly mappable if not null
       MNote     4,'DCU — Unknown character at position &J converte*
           d to blank.'
&C     SetC  '20'              Unicode blank
.*
.Z2    ANop  ,
&R     SetC  '&R.00&C'         Add new character to end
&J     SetA  &J+1
       AIf   (&J lt K'&A).Z    Repeat for all internal characters
.*-------------------------------------------------------------*
.*    Generate the requested Unicode constant                  *
.*-------------------------------------------------------------*
&L     DC    X'&R'
       MEnd
```

Fig. 7